APPENDIX A: XML SCHEMA FOR MLD

This appendix describes illustrative XML schema used in the MLD definition.

LANGUAGE

The top-level element in the MLD definition.

Syntax:

<LANGUAGE *attribute-list*>

Description of the default element (DEFAULT)

Description of the root element (ROOT)

Zero or more element type descriptions (ELEMENT)

</LANGUAGE>

Attributes:

- **identifier** = "*attribute name*". The name of the attribute that is used as the element ID in the input XML file. Every element in the input file must have a unique element ID, i.e. it must have this attribute with a unique value. This attribute is required.
- **name** = "*language name*". The optional name of the markup language.

DEFAULT

Describes a default element. This description will be applied to all elements that have not been described by ELEMENT tag, excluding the root, which has its own tag.

Syntax:

<DEFAULT *attribute-list*>

One or more property descriptions (PROPERTY)

</DEFAULT>

Attributes:

- **rank** = "normal | low | ignore". The importance of the element in document recognition. The value are interpreted as follows:
    - **normal** – the element is considered moderately important in document recognition.
    - **low** – the element has a low importance in document recognition.
    - **ignore** – the element is not considered in document recognition. This is the default value.

ROOT

Describes the root element.

Syntax:

<ROOT>

One or more property descriptions (PROPERTY)

</ROOT>

ELEMENT

An element type description that instructs the CRM what properties to use for each type of element. The type of element is the element name. The inner content consists of one or more property definitions.

Syntax:

<ELEMENT *attribute-list*>

One or more property descriptions (PROPERTY)

</ELEMENT>

Attributes:

- **rank** = "necessary | high | normal | low | ignore". The importance of the element in document recognition. The values are interpreted as follows:
  - **necessary** – the element must be in the document.
  - **high** – the element has a high importance in document recognition.
  - **normal** – the element is considered moderately important in document recognition.
  - **low** – the element has a low importance in document recognition.
  - **ignore** – the element is not considered in document recognition. This is the default value.

PROPERTY

Specifies the property as being relevant to the element.

Syntax:

<PROPERTY *attribute-list* />

Attributes:

- **class** = "*class name*". The name of the class implementing the property. The class must exist in CRM classpath. If it does not exist, a default property always returning 0 will be used.
- **rank** = "necessary | high | normal | low". The rank of the property, from which the property weight will be generated. The values are interpreted as follows:
  - **necessary** – the element must have the property.
  - **high** – the element is very likely to have the property.
  - **normal** – the property gets a standard treatment.
  - **low** – the element is not very likely to have this property.

- A list of property attributes (with their values) is included in the PROPERTY element. The names of the attributes can be any legal XML identifier, except for identifiers "class", "rank", or "weight".

5

APPENDIX B: XML SCHEMA FOR EDD

This appendix describes illustrative XML schema used in the EDD definition.

CRM:DESCR

The top-level element in EDD declaration. Provides the CRM with the ID of the document.

Syntax:

<CRM:DESCR *attribute-list*>

Zero or more element declarations (CRM:EL)

</CRM:DESCR>

Attributes:

- **docid** = "*document ID*". The required unique document ID.
- **markup** = "*the name of the MLD* ". The name of the MLD file to be used with this EDD. This attribute is required.

CRM:EL

Declares an element as element of interest. May specify optional content ID, which would mean that the content of this element can be obtained from the CRM.

Syntax:

<CRM:EL *attribute-list*>

Zero or more used-specified element properties (CRM:PROP)

</CRM:EL>

Attributes:

- **id** = "*element ID*". The required unique element ID.

- **cid** = "*content ID*". The optional content ID.
- **rank** = "necessary | high | normal | low | ignore". The importance of the element in document recognition. The values are interpreted as follows:
    - **necessary** – the element must be in the document.
    - **high** – the element has a high importance in document recognition.
    - **normal** – the element is considered moderately important in document recognition.
    - **low** – the element has a low importance in document recognition.
    - **ignore** – the element is not considered in document recognition. This is the default value.

CRM:PROP

Describes a property of element.

Syntax:

<CRM:PROP *attribute-list* />

Attributes:

- **class** = "*class name*". The name of the class implementing the property. The class must exist in CRM class path. If it does not exist, a default property always returning 0 will be used.
- **rank** = "necessary | high | normal | low". The rank of the property, from which the property weight will be generated. The values are interpreted as follows:
    - **necessary** – the element must have the property.
    - **high** – the element is very likely to have the property.
    - **normal** – the property gets a standard treatment.
    - **low** – the element is not very likely to have this property.

- A list of property attributes (with their values) is included in this PROP element. The names of the attributes can be any legal XML identifier, except for identifiers "class", "rank", or "weight".

APPENDIX C: XML SCHEMA FOR IDD

This appendix describes illustrative XML schema used in the IDD definition.

CRM:DOC

The top-level element in document description.

Syntax:

<CRM:DOC *attribute-list*>

Zero or more of internal element descriptions (CRM:E)

</CRM:DOC>

Attributes:

- **id** = "*document ID*". The unique ID of the document.
- **markup** = "*the MLD file name*". The name of the MLD file from which the IDD was generated.
- **threshold** = "*threshold value*". The minimum threshold for document recognition.

CRM:C

The internal description of an element.

Syntax:

<CRM:C *attribute-list*>

One or more properties (CRM:P)

</CRM:C>

Attributes:

- **weight** = "*presence weight*". The presence weight of the element.

- **threshold** = "*the threshold value*". The threshold to which compare the content recognition score.

CRM:P

A description of property.

Syntax:

<CRM:P *attribute-list* />

Attributes:

- **class** = "*class name*". The name of the class implementing the property.
- **weight** = "*property weight*". The weight of the property.
- A list of property attributes (with their values) is included in this PROP element. The names of the attributes can be any legal XML identifier, except for identifiers "class", "rank", "threshold", or "weight".

APPENDIX D: PROPERTIES

OVERVIEW

This subsection provides a general overview of properties used in the CRM.

Expressiveness of Properties

Any Boolean property about the document, element, or element's attribute can be expressed as a property in the CRM, as long as it is computable by a Turing Machine given the document. This also covers some non-Boolean properties: those that can evaluate to a number from 0 to 1, inclusive. Every property has two methods with an arbitrary implementation (whatever can be done in Java):

- **Property constructor** – this method computes some property attributes that will be stored in the IDD for later comparison/evaluation using another document. The constructor has the entire document available to itself.

- **Property evaluation function** – a function that returns the value to which the property evaluates. The parameter is an element in the document. However, the function can have any number of values (property attributes) in which context it will be evaluated. Implicitly, the evaluation function has access to all elements that have been found so far on the document (for binary properties), and to the entire document being recognized.

The following sections provide some examples of properties.

Attribute Properties

Properties can describe attributes of an element, in many ways. For example, property can state that an attribute's value is equal to some string, that it is very similar to some string (it is left to implementer of the evaluation function to decide what "similar" means), or that it matches some regular expression. These 3 cases are discussed in detail below.

The string equality can be a straightforward implementation, but the comparison may be case-sensitive or case-insensitive, depending on the language, element, and the attribute. For example, the "method" attribute of XHTML's "form" element can have two values: "get" and "post", but "GET" is the same as "get" or "GeT", so a case-insensitive comparison may need to be used. Moreover, the value can be a list separated by some delimiter, in which case the comparison must be done on two lists, not two strings. Finally, it can be a numeric expression, which may need to be evaluated, e.g. values "001" and "1" are different as strings, but identical as numbers. In practice, there will be a variety of properties that compare attributes for equality. Which property to use will depend on the attribute and language.

For some languages, including XHTML, the attribute equality property shall be listed in the MLD carefully. In other words, it is better be specified by the user. The reason is that in many cases the CRM cannot be sure that the attribute value will always stay the same. However, in some cases it can be assumed that the value won't change, e.g. an "id" attribute of an element in XHTML document.

The string similarity properties are commonly used in the MLD and EDD. There is a significant number of similarity properties, basically, one property per attribute name. To illustrate, consider strings "3" and "2". If they are compared using string similarity, then the result (using common sense) shall be the same as for pair "3" and "9". However, if they are compared as numbers, then the similarity between "3" and "2" is better, since the numbers are closer to each other. As another example, consider

similarity between URL's. If they are compared as generic strings, then every character has the same significance. However, a higher significance shall be placed on domain name and protocol name. Furthermore, the property that is designed specifically for url comparison shall know that "domain/name" is the same as "domain/./name" and "domain/../domain/name". The exact nature of how comparison will be done will be determined from the description of the attribute in the language documentation.

The regular expression properties are inherently Boolean in nature (otherwise, they require a cumbersome implementation). Thus, they can only be provided by the user through the EDD. As an XHTML example, the user can say that the "href" attribute of a link must match some regular expression, which is alone may be sufficient to recognize the link.

Inner Text Properties

Similarly to the attributes, there can be inner text properties which can be tested for equality, similarity (mostly as strings), or regular expression match. In addition, a property of inner text can have an exclusion set, which tells in which elements not to descend when computing the inner text. For XHTML example, a link's inner text can have <font size="=1"> element, which may be present or may be absent and whose inner text shall not be considered when considering inner text of entire <a> element. Thus, "font" can be in the exclusion list, which means that CRM would not recursively descend into "font" elements and get their inner text.

For XHTML, the MLD should take minimal consideration of the inner text. Perhaps, only the inner text of "title" element can be compared for similarity. In general, the title seldom changes for a document. A user can state in EDD that the title is equal to some user-supplied string or matches some regular expression.

Moreover, there can be properties for inner text when it is considered in many formats. For example, there may be a property that states that inner text is a number, a stock quote, or a single word.

It is also possible to include inner elements as well, which then makes the property not specific to the inner text, but rather to the more general inner content. For example, a fraction in XHTML may consist of digits and "sup" and "sup" elements, separated by "/".

Inner Structure Properties

Properties can also describe the inner structure of the element. For example, a count of "a" elements, a count of "tr" elements, etc in XHTML document is useful to recognize a "table" element. Moreover, properties can have some representation of the inner structure (e.g. table structure, etc) as one of their (computable) attributes and then compare that structure to the structure of the element in question.

Outer Structure Properties

Similarly to inner content properties, there can be outer content properties, that can describe the rest of the document, especially the path from the root to the element.

Binary Properties

The property evaluation function has always access to all the elements that have already been found on the document. Hence, there can be properties that refer to some other element in their evaluation function. For the purposes of CRM, the main concern is with "binary" properties, those properties that refer to a single element that has already been

found. Examples of such properties are: "is descendant of", "is sibling of", and "is parent of", etc.

Recommended Properties for User

The IDT can use the following properties:

- **Attribute value match against some regular expression** – the pattern must be supplied by the user.

- **Attribute value equality** – the value to compare must be the current value of the attribute, which the user can see through some UI

- **Attribute value similarity** – if the value changes and the user is unable to give a regular expression pattern, then the user can say that the value will not significantly change. This is not so true for general strings, but may be tru for numbers (and, perhaps, URLs).

- **Inner content** – same things can be applied to inner content as to attribute values.

- **Invariant elements** – if some element *A* in inner content of the element of interest *X* is known to always stay the same, then it can be specified as "invariant" in terms of its own properties and then the element of interest can be given a binary property which tells that *X* is an ancestor of *A*. Clearly, to use that property, *A* must be located before *X*, but that is guaranteed if *A* is in fact invariant (always present on the document) and its weight ("rank" attribute in the EDD) is higher than that of *X*

- **Inner count** – a user can say that the inner count of some elements in the element of interest is between some 2 numbers, given by the user. For example, "a table has at least 20 links" or "a cell has between 5 and 6 paragraphs". By keeping the lower and the upper limit the same, one can achieve equality condition.

- **Inner/Outer structure** – a user can be asked what is more important in element recognition, the structure of its inner content or the path from the root to the element.